



Norwegian University of Science and Technology
Engineering
The Department of Computer and Information Science

TDT4160

DATAMASKINER OG DIGITALTEKNIKK

EKSAMEN

2. DESEMBER, 2015, 09:00–13:00

Kontakt under eksamen:

Gunnar Tufte 974 02 478

Tillatte hjelpemidler:

D.

Ingen trykte eller håndskrevne hjelpemidler er tillat.

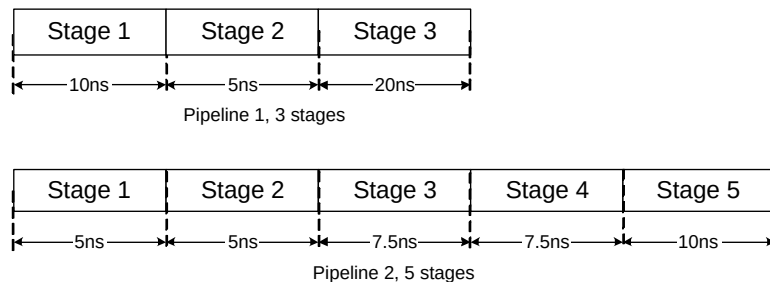
Enkel godkjent kalkulator er tillatt.

Målform:

Bokmål

OPPGAVE 1: OPPSTART, LITT AV HVERT (25 %)

- a. En ISA har tre alternative mikroarkitekturer. Alternativ 1 er utan samlebånd (pipeline) med en klokkeperiode på $40ns$. Alternativ 2 er med et samlebånd med tre trinn. Alternativ 3 er med et samlebånd på 5 trinn. Alternativ 2 og 3 er skissert i figur 1. Er det mulig å øke klokkefrekvensen i mikroarkitekturerne i alternativ 2 og 3? Oppgi klokkefrekvens for de to alternativene.



Figur 1: Samlebånd for alternativ 2 og 3.

- b. En prosessor har et enkelt minnesystem med et hovedminne (RAM) med aksesstid på $0.5\mu s$. I et forsøk på å skalere opp systemet og øke ytelsen ønsker man å doble minnestørrelsen og minske gjennomsnittlig aksesstid. I et forsøk blir hovedminnet byttet ut med et dobbelt så stort minne med aksesstid på $2\mu s$ og et nivå hurtigbuffer (cache). Hurtigbuffer har en aksesstid på $0,025\mu s$ og trefforholdstall for systemet (hit ratio) er 90 %

Hva er gjennomsnittlig aksesstid for det nye minnesystemet (med et nivå hurtigbuffer)?

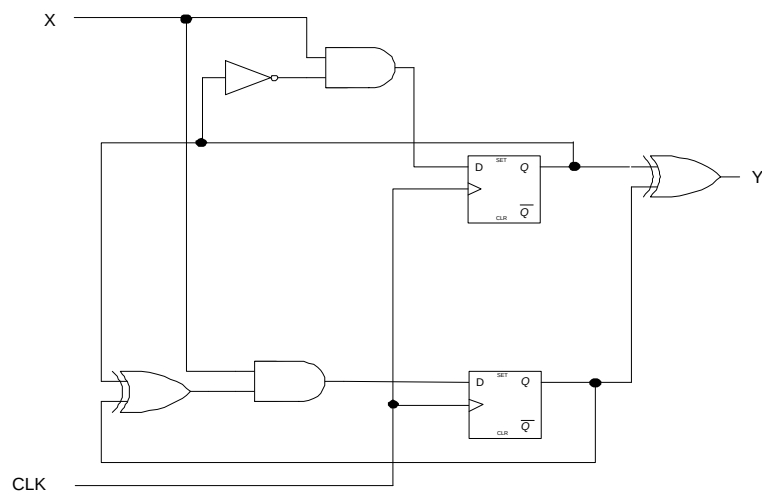
- c. Hva kjennetegner en superscalar prosessor? Forklar kort.
- d. Hva kjennetegner en SIMD-prosessor? Forklar kort.
- e. Er grad av ILP definert på ISA-nivå?

I et innvevd system (embedded system) benyttes det en mikrokontroller. Figur 2 viser det eksterne bussgrensesnittet for mikrokontrolleren. Det er en ROM-brikke for program og to RAM-brikker. Alle enhetene benytter et aktivt lavt (logisk "0") CS (Chip Select)-signal. RAM-brikkene har en aksestid på $0.1\mu s$ ROM-brikken har en aksestid på $1\mu s$.



- Page 3 of 19

c. Figur 3 viser en FSM.



Figur 3: FSM.

- i) Er dette en Moore eller Mealy FSM? Forklar.
- ii) Fullfør tabellen vist under. Bruk formatet som er vist.

Q0 Q1		Q0(nxt) Q1(nxt)	
		X = 0	X = 1
0	0		

Figur 4: FSM tabell format.

- iii) Når er utgangen $Y = "1"$?

OPPGAVE 3: MIKROARKITEKTUR OG MIKROINSTRUKSJONER (25 % (5 % PÅ A, 10 % PÅ B OG C))

Bruk vedlagte diagram i figur 6, figur 7, figur 8 og figur 9 for IJVM til å løse oppgavene.

- a. i) Er det noen av registrene i IJVM som ikke er tilgjengelig for programmereren?
ii) Hvilken funksjon har MAR-registeret og MDR-registeret? Forklar kort.
- b. i) Lag mikroinstruksjon(er) som utfører funksjonen: kopier TOS til MAR ($MAR = TOS$) for mikroarkitekturen i figur 6.
ii) Lag mikroinstruksjon(er) som utfører funksjonen: summerer innholdet i LV, MDR og PC til PC ($PC = LV + MDR + PC$) for mikroarkitekturen i figur 6. Hvordan vil LV og MAR bli påvirket av løsningen din?

Du trenger ikke ta hensyn til Addr- og J-feltene. Oppgi korrekt bit-verdi for ALU-, C-, Mem- og B-feltene. – Se figur 7.

- c. IJVM-registrene i figur 6 er satt til følgende verdier:

"SP": hex(0001),
"LV": hex(0002),
"CPP": hex(0003),
"TOS": hex(0004),
"OPC": hex(0005),
"H": hex(FF0A).

Hva er innholdet i H-registeret og OPC-registeret etter at følgende mikroinstruksjoner har blitt utført? Oppgi svaret i hex-format.

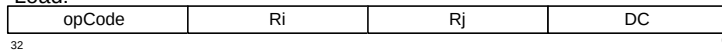
- 1: ALU: 010100, C: 100000000, Mem: 000 og B: 0110
- 2: ALU: 011100, C: 101000000, Mem: 000 og B: 0111
- 3: ALU: 001100, C: 001000000, Mem: 000 og B: 1000

OPPGAVE 4: INSTRUKSJONSSETT ARKITEKTUR (ISA)(25 % ((5 % PÅ A, B,C OG 10 % PÅ D))

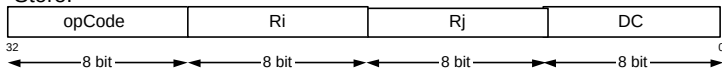
En avært enkel prosessor har en load, en store, 8 ALU-instruksjoner og noen spesialinstruksjoner, inkludert NOP-instruksjonen og to flytkontrollinstruksjoner (flow control instructions). Instruksjonsformatet for instruksjonene er vist i figur 5. Alle register og bussar er 32-bit. Det er 16 generelle register tilgjengelig. Prosessoren har en Harvard architecture.

Load/store:

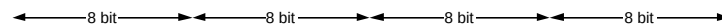
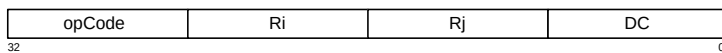
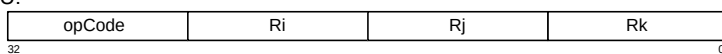
Load:



Store:

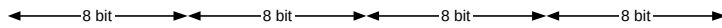
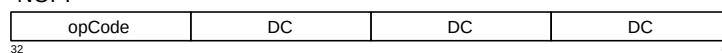


ALU:

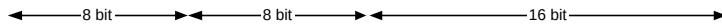
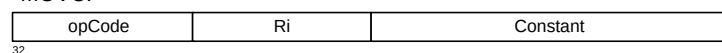


Spesial:

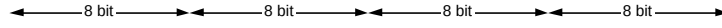
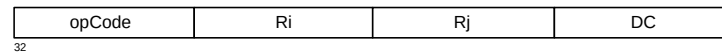
NOP:



MOVC:

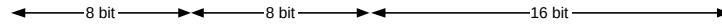
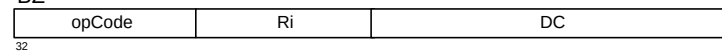


CP:

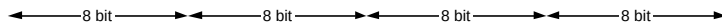
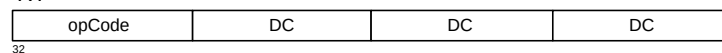


Flow control:

BZ



RT



Ri, Rj, Rk: any user register, R0 - R15

DC: Don't care: any data or memory location

Figur 5: Instruction formats.

Instructions set:

LOAD: Load data from memory.

load Ri, Rj Load register Ri from memory location in Rj.

STORE: Store data in memory.

store Ri, Rj Store register Ri in memory location in Rj.

ALU: Data manipulation, register-register operations.

ADD Ri, Rj, Rk ADD, $Ri = Rj + Rk$. Set Z-flag if result =0.

NAND Ri, Rj, Rk Bitwise NAND, $Ri = \overline{Rj \cdot Rk}$. Set Z-flag if result =0.

OR Ri, Rj, Rk Bitwise OR, $Ri = Rj + Rk$. Set Z-flag if result =0.

INV Ri, Rj Bitwise invert, $Ri = \overline{Rj}$. Set Z-flag if result =0.

INC Ri, Rj Increment, $Ri = Rj + 1$. Set Z-flag if result =0.

DEC Ri, Rj Decrement, $Ri = Rj - 1$. Set Z-flag if result =0.

MUL Ri, Rj, Rk Multiplication, $Ri = Rj * Rk$. Set Z-flag if result =0.

CMP, Ri, Rj Compare, Set Z-flag if $Ri = Rj$

Special: Misc.

CP Ri, Rj Copy, $Ri < -Rj$ (copy Rj into Ri)

NOP Waste of time, 1 clk cycle.

MOVC Ri, constant Put a constant in register $Ri = C$.

Flow control: Branch.

BZ, Ri Conditional branch on zero, $PC = Ri$.

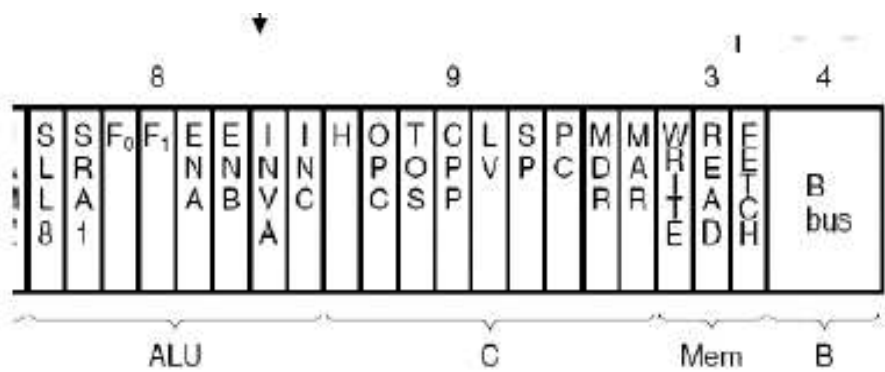
RT Return, return from branch.

Ri, Rj and Rk: Any user register.

DC: Don't care.

- a. Hva slags type instruksjoner er:
- BZ?
 - RT?
- b. Hva slags type adresseringsmodi (addressing modes) bruker:
- MOVC?
 - CP?
- c. Kan LOAD- og STORE-instruksjonene adressere hele adresserommet (32-bit adressebusser)?
- d. Følgende psaudokode er en del av et større program. Svar på spørsmålene ut i fra tilgjengelig informasjon.
- MOVC R1, 0x00 55;
 - MOVC R2, 0x00 AA;
 - STORE R1, R2;
 - ADD R1, R1, R3;
 - STORE R1, R3;
 - LOAD R1, R2
 - INV R3, R3;
 - CMP R1, R2;
 - BZ R2;
- Hva slags adresse vil BZ hoppe til hvis betingelsene for et hopp er til stede?
 - Vil psaudokode føre til at Z-flagget blir aktivert slik at betingelsene for et hopp er til stede? Begrunn svaret.

IJVM appendix



B bus registers

0 = MDR	5 = LV
1 = PC	6 = CPP
2 = MBR	7 = TOS
3 = MBRU	8 = OPC
4 = SP	9-15 none

Figur 7: Microinstruction format (IJVM).